Computer Vision Group
TUM Department of Informatics
Technical University of Munich

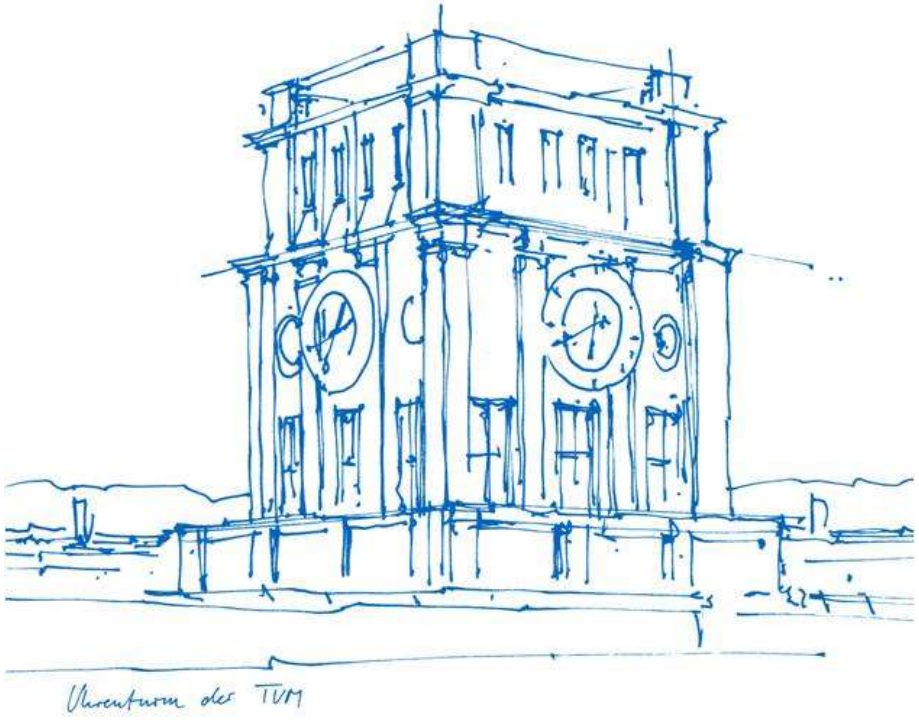
Automatically Differentiable Dense Visual Dynamic SLAM for RGB-D Cameras

Master's Thesis – Final Presentation

Jingkun Feng, January 18th , 2024, Munich

Supervisor: Prof. Dr. Daniel Cremers

Advisor: M.Sc. Mariia Gladkova



Outline

- Introduction
- Related Works
- Proposed Solution
- Evaluation
- Discussion and Future Work
- Summary

Introduction

- ∇SLAM [1] provides insights into automatically differentiable SLAM
  - Express SLAM as a differentiable function
  - Enables end-to-end learning in conjunction with downstream tasks

- Realistic applications are in a dynamic world, but most SLAM systems assume a static environment

Figure 1. AR application

Figure 2. Robot manipulation

- This project **aims** to propose a new **differentiable RGB-D SLAM** framework for dynamic environments

[1] K. M. Jatavallabhula, S. Saryazdi, G. Iyer, and L. Paull, “gradSLAM: Automagically differentiable SLAM.”

Related Work

- ∇SLAM [1]
  - Proposed differentiable alternatives for non-differentiable components
  - ✓ Fully differentiable
  - × No trainable parameters
  - × Performance bounded by conventional counterparts

- Direct method applying feature-metric representation
  - GN-Net [2] and “Deep Probabilistic Feature-metric Tracking” [3]
  - ✓ Enlarger convergence basin

- DirectTracker [4]
  - 3D multi-object tracking using Direct Image Alignment (DIA)
- MaskFusion [5]
  - Combining Mask R-CNN [6] and geometric segmentation
  - Inspired the proposed moving object segmentation method

Proposed Solution

System Overview

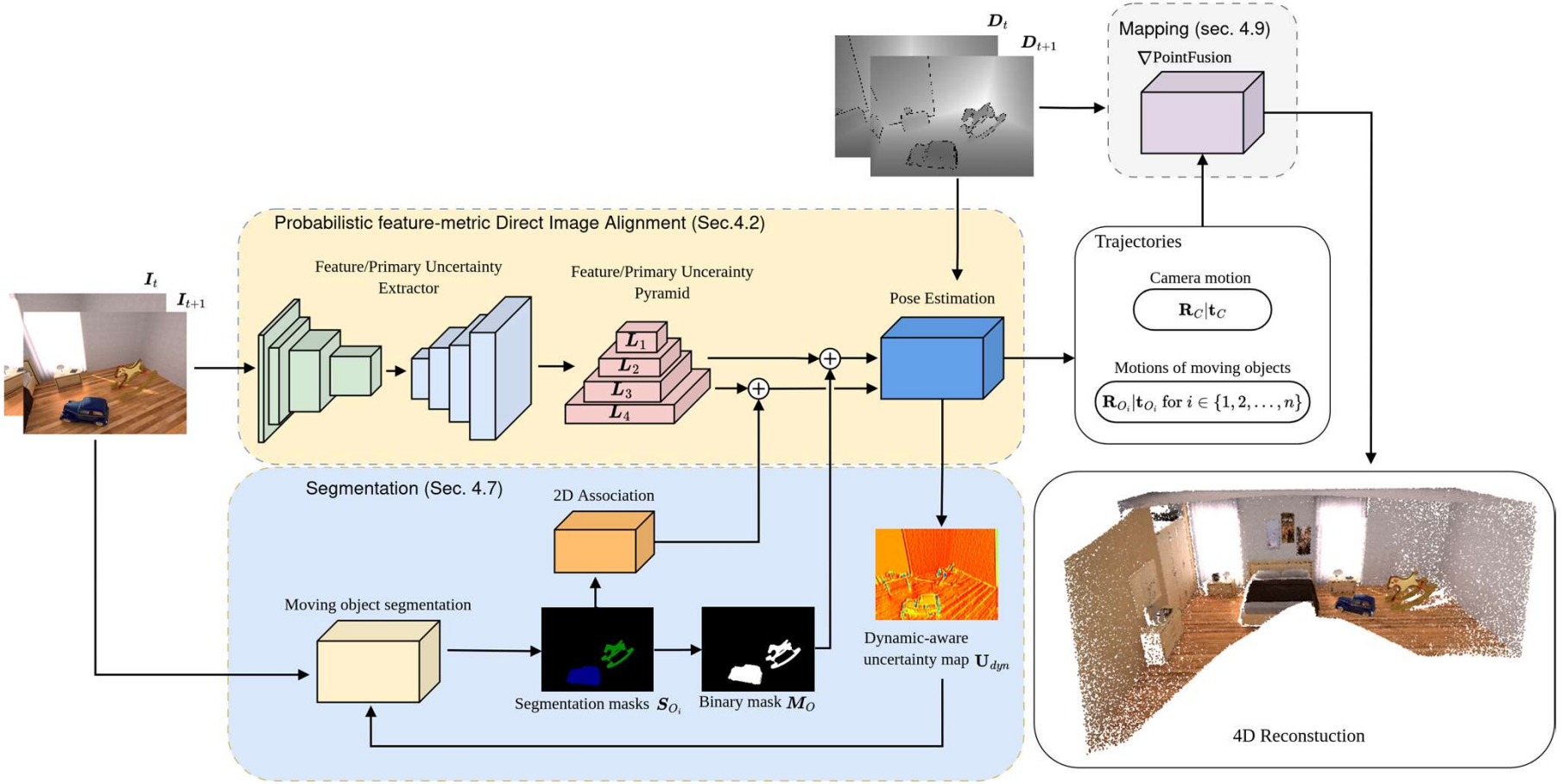


Figure 5. Computational graph of Direct Image Alignment

Proposed Solution – Visual Odometry

From ICP to Direct Image Alignment

- ∇SLAM employs ICP
  - ✗ Frame-to-model registration is computationally costly
  - ✗ Fully relies on depth input
  - ✗ Easily fails at repetitive or symmetric geometric features

Figure. Failure of ICP

- Direct Image Alignment
  - ✓ Frame-to-frame alignment is fast and yet accurate
  - ✓ Can be extended by combining intensity and depth errors

$$E_{track} = \min_{\xi_{ij} \in se(3)} (E_{photo} + \omega E_{geo})$$

$$r_I = \mathbf{I}_i(\mathbf{x}') - \mathbf{I}_j(\mathbf{x})$$

$$r_D = \mathbf{D}_i(\mathbf{x}') - \left[\mathbf{T}_{ij} \cdot \Pi^{-1}(\mathbf{x}_j)\right]_Z$$

Figure. Reconstruction using DIA

Proposed Solution – Visual Odometry

From convention DIA to probabilistic feature-metric DIA

- Probabilistic feature-metric performs over features learned by a CNN
- At each pyramid level l, CNN extracts
  - a D_l-dimensional feature map $\mathbf{F}^{\mathrm{l}} \in \mathbb{R}^{W_l \times H_l \times D_l}$
  - a pixel-wise uncertainty map $\boldsymbol{\sigma}^{l} \in \mathbb{R}^{W_l \times H_l}$
- Formulate an uncertainty-normalized feature difference following [3]

$$\mathbf{r}_F = \mathbf{F}_i(\mathbf{x}') - \mathbf{F}_j(\mathbf{x})$$

$$\mathbf{r}_{PF} = \frac{\mathbf{r}_F}{\sigma_F} = \frac{\mathbf{F}_i(\mathbf{x}') - \mathbf{F}_j(\mathbf{x})}{\sqrt{\sigma_i^2(\mathbf{x}') + \sigma_j^2(\mathbf{x})}}$$

- Provides larger and smoother convergence basin for DIA

Proposed Solution – Visual Odometry

Deep M-Estimator

- M-Estimator is more robust to outliers than trivial least squares

$$\underset{\mathbf{T}_{ij} \in \mathcal{SE}(3)}{\arg\min}\, \mathbf{r}^T \mathbf{W} \mathbf{r}$$

- Inspired by “Taking a Deeper Look at the Inverse Compositional Algorithm” [7], the diagonal weight matrix is learned using a fully convolutional network

$$\mathbf{W}_\theta = \phi_M\left(\mathbf{F}_i(\mathbf{x}'_k), \mathbf{F}_j(\mathbf{x}), \mathbf{r}_k\right)$$

- Solved using the Levenberg-Marquardt (LM) algorithm

$$(\mathbf{J}^T \mathbf{W} \mathbf{J} + \lambda\, \mathrm{diag}(\mathbf{J}^T \mathbf{W} \mathbf{J}))^{-1} \Delta\xi = \mathbf{J}^T \mathbf{W} \mathbf{r}$$

- $\boldsymbol{W}_\theta$ serves as the prior for our uncertainty-based segmentation approach

Proposed Solution – Visual Odometry
Differentiable LM Algorithm

- Original LM algorithm is non-differentiable

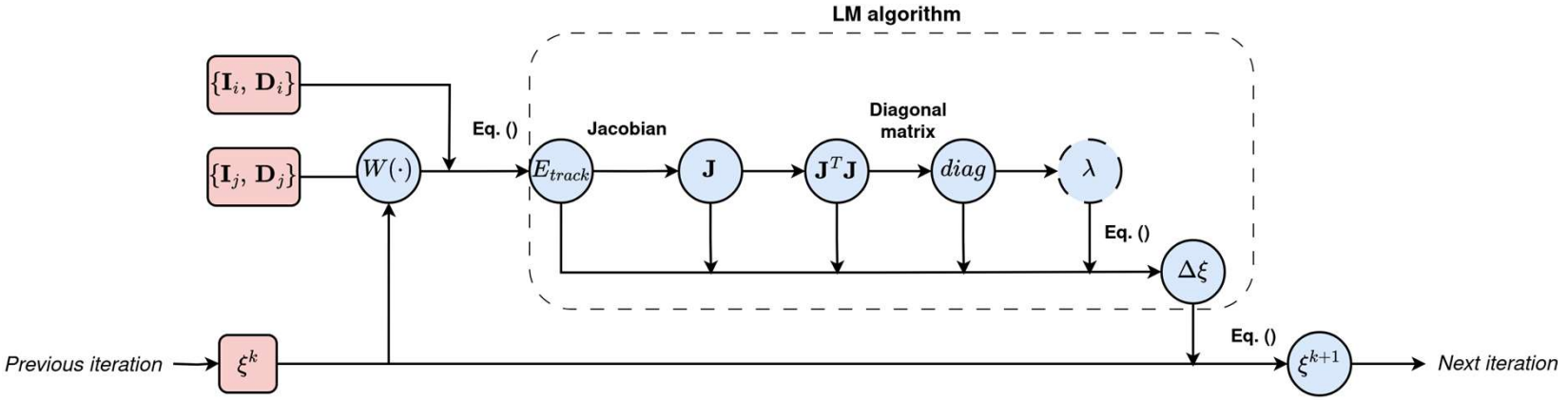


Figure 5. Computational graph of DIA using LM

- Following previous works [7] [8], we predict the damping factor λ_θ

$$\lambda_\theta = \phi_{LM}\left(\mathbf{J}^T\mathbf{W}\mathbf{J}, \mathbf{J}^T\mathbf{W}\mathbf{r}\right)$$

- And fix the number of update steps

Proposed Solution – Moving Object Segmentation

Mask R-CNN [6] refined by Geometric Segmentation

- Inspired by MaskFusion [5]
- Use Mask R-CNN to predict the prior segmentation
  - Focus on selected classes, such as humans, animals, and cars
- Refine using geometric segmentation
  - Investigate the depth discontinuity ϕ_d

$$\phi_d = \max_{i \in \mathcal{N}} |(\mathbf{v}_i - \mathbf{v}) \cdot \mathbf{n}|$$

  - and object surface concavity ϕ_c

$$\phi_c = \max_{i \in \mathcal{N}} \begin{cases} 0 & \text{if } (\mathbf{v}_i - \mathbf{v}) \cdot \mathbf{n} < 0 \\ 1 - (\mathbf{n}_i \cdot \mathbf{n}) & \text{otherwise} \end{cases}$$

  - Compute an edge map using $\phi_d + \kappa \phi_c > \theta$
  - Generate over-segmentation results by applying connected-components analysis (CCA) on edge map

Proposed Solution – Moving Object Segmentation

Uncertainty-based Segmentation using FastSAM [9]

- The first segmentation method does not distinguish instances
- Uncertainty map encodes dynamic pixels
- FastSAM [] can produce segmentation using prompt inputs

Generate prompts based on uncertainty maps to segment moving objects using FastSAM

Proposed Solution – Moving Object Segmentation

Uncertainty-based Segmentation using FastSAM [9]

- Train the model in two stages
  - Stage 1: using only static scenes to train uncertainty caused by illumination changes and sensor noise
  - Stage 2: using only dynamic scenes to train uncertainty solely caused by moving objects
- Generate prompt points by performing MaxPool and AvgPool on the uncertainty map
- Control the number of points within a limited set. For each point, generate a segmentation
- Merge segmentations that share large overlaps

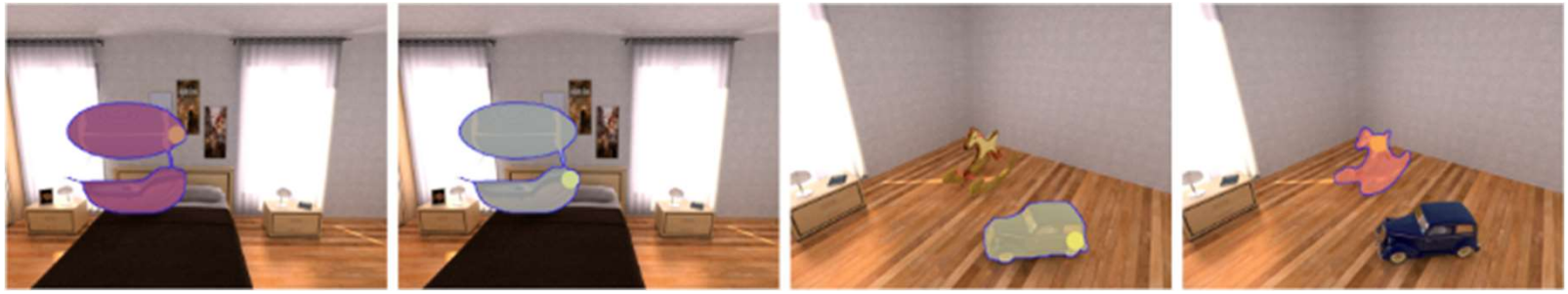


Figure 5. Segmentation results of images from the Co-Fusion [] dataset

Proposed Solution – 2D Association

- Conventional Hungarian Algorithm is non-differentiable
- Adapt the Hungarian Network (Hnet) proposed in [10]
- Hnet estimates the association matrix given the distance matrix

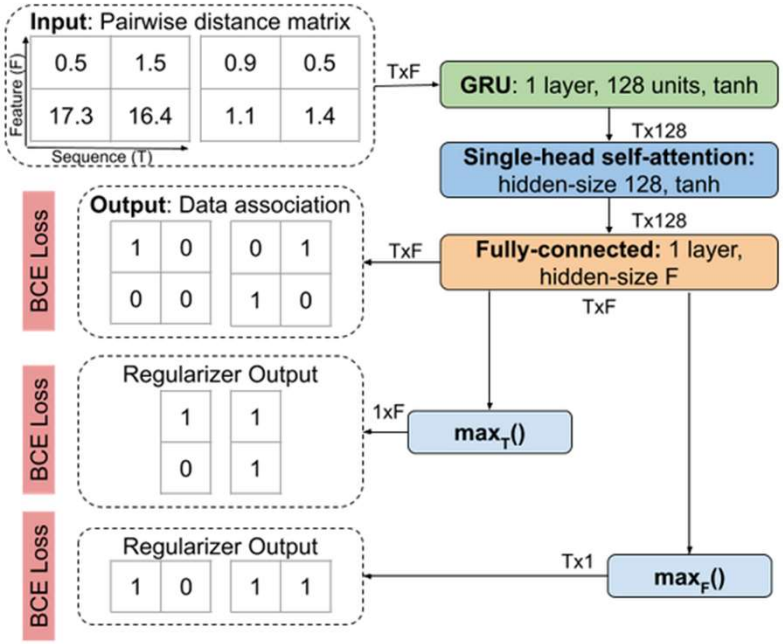


Figure 5. Illustration of Hnet [10]

[10] Adavanne, S. et al. Differentiable Tracking-Based Training of Deep Learning Sound Source Localizers.

Proposed Solution – Moving Object Tracking

- Formulate DIA only considering the pixels within the associated segmentation mask

$$E_{O_i}(\xi_{O_r O_l}^{C_r}) = \sum_{\mathbf{x} \in \mathcal{S}_i} \left(\mathbf{r}_{PF}^T \mathbf{W}_{PF} \mathbf{r}_{PF} + \mathbf{r}_D^T \mathbf{W}_D \mathbf{r}_D \right)$$

- For objects of small size or at a further distance
  - Trivial coarse-to-fine Optimization leads to inaccurate estimates or failure
  - Inspired by DirectTracker [4], scaling levels are adapted based on the size of objects

Proposed Solution

System Overview

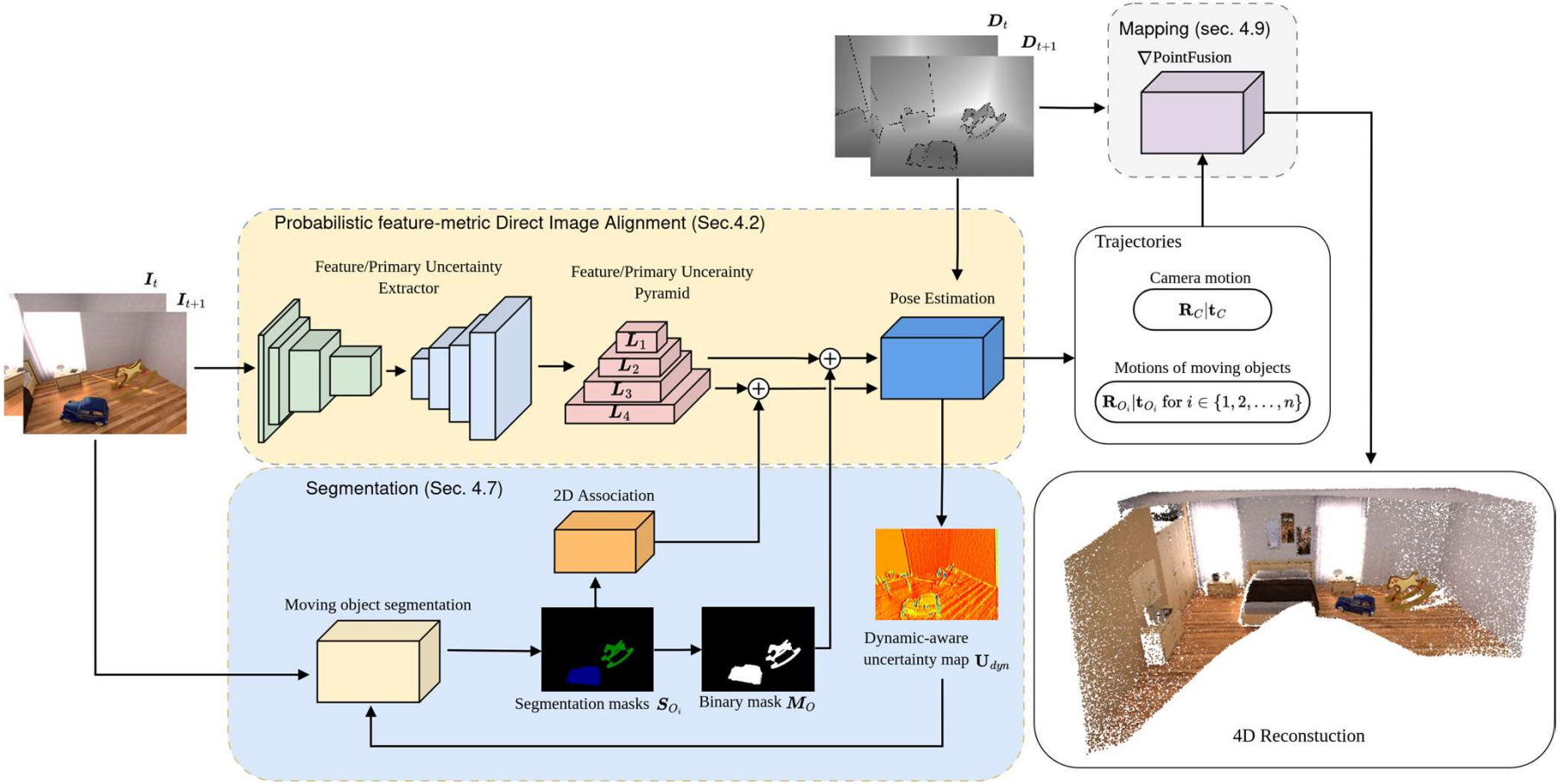


Figure 5. Computational graph of Direct Image Alignment

Proposed Solution
Implementation Details

- Full system is implemented in Python by extending ∇SLAM [1]
- Differentiability is guaranteed by using PyTorch
- 16M learnable parameters in our implementation
- Trained using RGB data from TUM RGB-D and Bonn Dynamic RGB-D datasets
- Supervised through 3D End-Point-Error:

$$L = \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{L}} \sum_{\mathbf{p}_j \in \mathcal{P}} \| \exp(\hat{\xi_{ij}}) \mathbf{p}_j - \exp(\xi_{ij}) \mathbf{p}_B \|^2$$

Evaluation
Datasets and Metrics

- Evaluation is conducted on synthetic and real datasets, including
  - ICL-NUIM [11]
  - TUM RGD-D [12]
  - Bonn Dynamic RGB-D [13]
  - Co-Fusion [14]

- Object tracking is evaluated only on the Co-Fusion dataset

- The performance is assessed using
  - absolute trajectory error (ATE)
  - relative pose error (RPE_{trans} and RPE_{rot})
  - Intersection over union (IoU) on the Co-Fusion sequences

Evaluation

Evaluation Results – Static Environment (ICL-NUIM)

	ColorICP[79]	RGB-D VO[78]	∇SLAM[1]	Ours	DROID[6]
living_room_traj0	0.2802	0.3603	**0.0954**	0.1281	0.6369
living_room_traj1	0.0171	0.5951	**0.0284**	0.0926	0.2954
living_room_traj2	0.1327	0.2931	**0.0334**	0.1008	0.8094
living_room_traj3	0.8168	0.8524	**0.3784**	0.4532	0.6545
office_traj0	0.2047	0.1710	0.0446	**0.0191**	0.5958
office_traj1	0.4062	0.5366	0.9921	**0.1360**	0.4122
office_traj2	0.2827	0.2289	0.0339	**0.0473**	1.0089
office_traj3	**0.0572**	0.2289	0.9553	0.1084	0.7044
Mean	0.2747	0.4083	0.3202	**0.1357**	0.6397

Table 1: Absolute Trajectory Error (RMS) on the ICL-NUIM dataset

Ours delivers the best average result over all sequences.

Evaluation

Evaluation Results – Static Environment (TUM RGB-D)

	Non-leanring-based			Learning-based	
	ColorICP [79]	RGB-D VO [78]	∇SLAM [1]	Ours	DROID [6]
Average	0.5428	0.9471	0.8823	0.3824	**0.3640**

Table 2: Absolute Trajectory Error (RMS) on the static sequences of the TUM RGB-D dataset

	Non-leanring-based						Learning-based			
	ColorICP		RGB-D VO		∇SLAM		Ours		DROID	
	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{rot}
Average	0.0170	0.8997	0.0311	18.8223	0.0204	1.1259	**0.0144**	**0.8089**	0.0600	3.8216

Table 3: Relative Pose Error (RMS) on the static sequences of the TUM RGB-D dataset

Ours outperforms conventional methods with a distinct advantage, while DROID performs slightly better than ours.

Evaluation

Evaluation Results – Dynamic Environment (TUM RGB-D)

		NL + NM			L + NM	L + M		NL + M		
		ColorICP [79]	RGB-D VO [78]	∇SLAM [1]	DROID [6]	Ours+GS	Ours+US	DS [48]	RF [74]	MF [45]
Slightly dynamic	fr3_s_static	0.0293	0.0224	0.0151	0.0067	0.0164	0.0095	**0.0064**	0.0110	0.0154
	fr3_s_xyz	0.2995	0.1718	0.1586	0.0288	0.1306	0.0963	**0.0146**	0.0315	0.0614
	fr3_s_rpy	0.0504	0.0581	0.0489	**0.0158**	0.0450	0.0482	0.0673	0.1996	0.0924
	fr3_s_halfsphere	0.2562	0.3808	0.2391	0.0217	0.1953	0.0676	**0.0196**	0.0387	0.0606
Highly dynamics	fr3_w_static*	0.0222	0.0223	0.0226	0.0169	0.0215	0.0226	**0.0067**	0.0170	0.5021
	fr3_w_xyz*	0.2839	0.2887	0.2806	0.0191	0.2736	0.2084	**0.0161**	0.0809	0.3457
	fr3_w_rpy	0.1553	0.1675	0.1628	0.0601	0.1566	0.0941	**0.0393**	0.2660	0.6605
	fr3_w_halfsphere	0.3817	0.4340	0.3834	0.0309	0.3601	0.2739	**0.0278**	0.0584	0.3906
	Average	0.1848	0.1932	0.1639	0.0250	0.1499	0.1026	**0.0247**	0.0879	0.2661

Table 4: Absolute Trajectory Error (RMS) on the dynamic sequences of the TUM RGB-D dataset

		NL + NM						L + NM		L + M				NL + M		
		ColorICP[79]		RGB-D VO[19]		∇SLAM[1]		DROID[6]		Ours+GS		Ours+US		DS[48]	RF[74]	MF[45]
Setting	Seq.	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{trans}	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{trans}	RPE_{trans}
Slightly dynamic	fr3_s_static	0.248	0.145	1.262	0.372	0.184	0.177	0.584	0.665	0.078	0.066	0.072	**0.033**	**0.012**	0.036	0.045
	fr3_s_xyz	0.422	0.299	1.554	0.465	0.337	0.371	1.072	3.220	0.330	0.201	0.187	**0.165**	**0.020**	0.047	0.054
	fr3_s_rpy	0.672	0.450	x	x	0.694	0.555	2.076	1.104	0.345	0.239	0.170	**0.198**	**0.131**	0.650	0.357
	fr3_s_halfsphere	0.583	0.461	1.008	0.426	0.437	0.528	2.953	3.161	0.341	0.248	0.177	**0.208**	**0.043**	0.048	0.084
Highly dynamics	fr3_w_static*	0.879	0.265	2.884	0.682	0.818	0.275	0.357	0.567	0.563	0.107	0.374	**0.084**	**0.010**	0.040	0.601
	fr3_w_xyz*	1.888	0.480	3.325	0.761	1.654	0.510	3.129	1.567	0.906	0.254	0.729	**0.220**	**0.023**	0.094	0.318
	fr3_w_rpy	1.910	0.563	24.767	2.959	1.773	0.652	2.667	3.417	1.061	0.313	0.926	**0.279**	**0.059**	0.280	0.570
	fr3_w_halfsphere	1.666	0.574	2.644	0.666	1.454	0.661	3.832	2.973	0.848	0.286	0.749	**0.257**	**0.039**	0.056	0.391
	Average	1.033	0.405	5.349	0.904	0.919	0.466	2.084	2.084	0.559	0.214	0.423	**0.180**	**0.042**	0.156	0.302

Table 5: Relative Pose Error (RMS) on the dynamic sequences of the TUM RGB-D dataset

Evaluation

Evaluation Results – Dynamic Environment (TUM RGB-D)

		NL + NM			L + NM	L + M		NL + M		
		ColorICP [79]	RGB-D VO [78]	∇SLAM [1]	DROID [6]	Ours+GS	Ours+US	DS [48]	RF [74]	MF [45]
Slightly dynamic	fr3_s_static	0.0293	0.0224	0.0151	0.0067	0.0164	0.0095	**0.0064**	0.0110	0.0154
	fr3_s_xyz	0.2995	0.1718	0.1586	0.0288	0.1306	0.0963	**0.0146**	0.0315	0.0614
	fr3_s_rpy	0.0504	0.0581	0.0489	**0.0158**	0.0450	0.0482	0.0673	0.1996	0.0924
	fr3_s_halfsphere	0.2562	0.3808	0.2391	0.0217	0.1953	0.0676	**0.0196**	0.0387	0.0606
Highly dynamics	fr3_w_static*	0.0222	0.0223	0.0226	0.0169	0.0215	0.0226	**0.0067**	0.0170	0.5021
	fr3_w_xyz*	0.2839	0.2887	0.2806	0.0191	0.2736	0.2084	**0.0161**	0.0809	0.3457
	fr3_w_rpy	0.1553	0.1675	0.1628	0.0601	0.1566	0.0941	**0.0393**	0.2660	0.6605
	fr3_w_halfsphere	0.3817	0.4340	0.3834	0.0309	0.3601	0.2739	**0.0278**	0.0584	0.3906
	Average	0.1848	0.1932	0.1639	0.0250	0.1499	0.1026	**0.0247**	0.0879	0.2661

Though ours performs better than non-learning static SLAM methods, DROID SLAM and other state-of-the-art dynamic SLAM methods are generally superior to ours.

Slightly dynamic	fr3_s_rpy	0.672	0.450	x	x	0.694	0.555	2.076	1.104	0.345	0.239	0.170	**0.198**	**0.131**	0.650	0.357
	fr3_s_halfsphere	0.583	0.461	1.008	0.426	0.437	0.528	2.953	3.161	0.341	0.248	0.177	**0.208**	**0.043**	0.048	0.084
Highly dynamics	fr3_w_static*	0.879	0.265	2.884	0.682	0.818	0.275	0.357	0.567	0.563	0.107	0.374	**0.084**	**0.010**	0.040	0.601
	fr3_w_xyz*	1.888	0.480	3.325	0.761	1.654	0.510	3.129	1.567	0.906	0.254	0.729	**0.220**	**0.023**	0.094	0.318
	fr3_w_rpy	1.910	0.563	24.767	2.959	1.773	0.652	2.667	3.417	1.061	0.313	0.926	**0.279**	**0.059**	0.280	0.570
	fr3_w_halfsphere	1.666	0.574	2.644	0.666	1.454	0.661	3.832	2.973	0.848	0.286	0.749	**0.257**	**0.039**	0.056	0.391
	Average	1.033	0.405	5.349	0.904	0.919	0.466	2.084	2.084	0.559	0.214	0.423	**0.180**	**0.042**	0.156	0.302

Table 5: Relative Pose Error (RMS) on the dynamic sequences of the TUM RGB-D dataset

Evaluation

Evaluation Results – Dynamic Environment (Bonn Dynamic RGB-D)

	NL + NM			L + M		NL+ M		
nce	ColorICP	RGB-D VO	SLAM	Ours+GS	Ours+US	**DS**	RF	MF
Average	0.155	0.243	0.176	0.147	0.134	**0.0946**	0.2317	0.1464

Table 6: Absolute Trajectory Error (RMS) on the dynamic sequences of the Bonn Dynamic RGB-D dataset

	NL + NM						L + M				NL + M		
	ColorICP		RGB-D VO		∇SLAM		Ours + GS		Ours + US		DS	RF	MF
	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{rot}	RPE_{trans}	RPE_{trans}	RPE_{trans}
Average	0.038	3.200	0.036	13.644	0.039	**2.626**	0.027	3.110	**0.025**	2.934	0.717	0.814	0.847

Table 7: Relative Pose Error (RMS) on the dynamic sequences of the Bonn Dynamic RGB-D dataset

Different from the results on TUM RGB-D, on this dataset, our method ranks second, holding on to the best method.

Table 5: Relative Pose Error (RMS) on the dynamic sequences of the TUM RGB-D dataset

Evaluation

Evaluation Results – Dynamic Environment (Co-Fusion)

		∇SLAM [1]			Ours			CoFusion [49]		
		ATE	RPE_{trans}	RPE_{rot}	ATE	RPE_{trans}	RPE_{rot}	ATE	RPE_{trans}	RPE_{rot}
room4	cam	0.3839	0.0211	0.8467	0.0267	0.0048	0.0977	**0.0141**	**0.0006**	**0.0003**
	ship	-	-	-	**0.0096**	0.1328	0.0150	0.0298 / 0.0217	**0.0170 / 0.0092**	**0.0255 / 0.0061**
	horse	-	-	-	1.0266	0.1949	3.6004	**0.0612**	**0.0061**	**0.6729**
	car	-	-	-	0.0687	0.0437	0.0620	**0.0032**	**0.0091**	**0.0009**
car4	cam	x	x	x	0.3737	0.0083	0.0210	**0.0273**	**0.0011**	**0.0009**
	truck1	-	-	-	0.2249	0.0106	0.7793	**0.0103**	**0.0010**	**0.0045**
	car2	-	-	-	0.1269	0.0281	0.4560	**0.0070**	**0.0033**	**0.0059**

Table 8: Evaluation results on the Co-Fusion dataset

Sequence	Object	Ours	Co-Fusion
room4	Airship	0.85	0.43 / 0.62
	Horse	0.78	0.48
	Car	0.88	0.88
car4	Truck	0.81	0.79
	Car	0.76	0.68

Table 9: Average IoU between the predicted masks and the ground truth labels

Evaluation

Evaluation Results – Dynamic Environment (Co-Fusion)

		∇SLAM [1]			Ours			CoFusion [49]		
		ATE	RPE_{trans}	RPE_{rot}	ATE	RPE_{trans}	RPE_{rot}	ATE	RPE_{trans}	RPE_{rot}
room4	cam	0.3839	0.0211	0.8467	0.0267	0.0048	0.0977	**0.0141**	**0.0006**	**0.0003**
	ship	-	-	-	**0.0096**	0.1328	0.0150	0.0298 / 0.0217	**0.0170 / 0.0092**	**0.0255 / 0.0061**
	horse	-	-	-	1.0266	0.1949	3.6004	**0.0612**	**0.0061**	**0.6729**
	car	-	-	-	0.0687	0.0437	0.0620	**0.0032**	**0.0091**	**0.0009**
car4	cam	x	x	x	0.3737	0.0083	0.0210	**0.0273**	**0.0011**	**0.0009**
	truck1	-	-	-	0.2249	0.0106	0.7793	**0.0103**	**0.0010**	**0.0045**
	car2	-	-	-	0.1269	0.0281	0.4560	**0.0070**	**0.0033**	**0.0059**

Table 8: Evaluation results on the Co-Fusion dataset

The proposed method shows relativey mediocre accuracy in motion estimation compared to Co-Fusion. However, we have better and more consistent segmentation results.

room4	Horse	0.78	0.48
	Car	0.88	0.88
car4	Truck	0.81	0.79
	Car	0.76	0.68

Table 9: Average IoU between the predicted masks and the ground truth labels

Evaluation

Evaluation Results – Dynamic Environment (Co-Fusion)

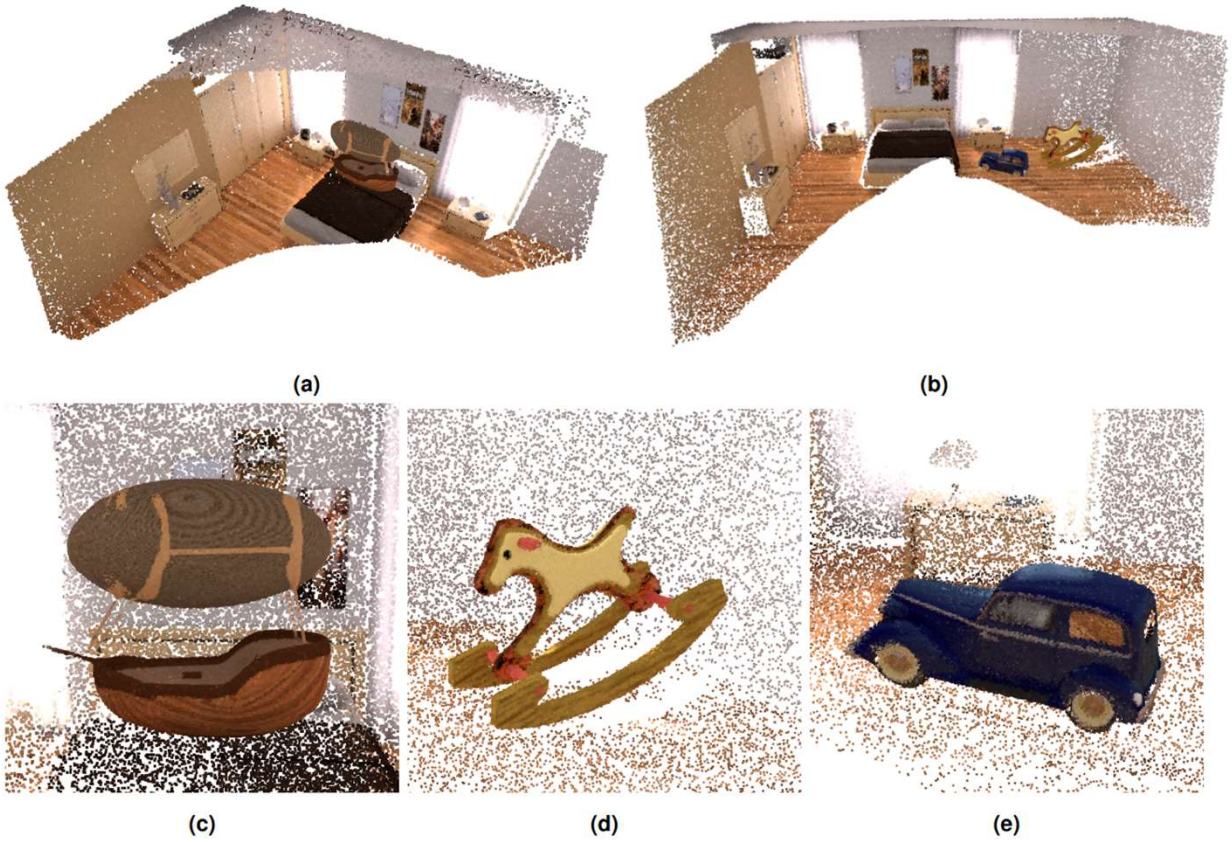


Table 8: Qualitative results on the room sequence of the Co-Fusion dataset

Discussion and Future Work

Ablation study

	F			F + P			F + U			F + P + U			Ours		
Sequence	ATE	RPE_{trans}	RPE_{rot}	ATE	RPE_{trans}	RPE_{rot}	ATE	RPE_{trans}	RPE_{rot}	ATE	RPE_{trans}	RPE_{rot}	ATE	RPE_{trans}	RPE_{rot}
fr1_360	0.1819	0.0094	1.1138	0.1568	0.0101	0.6711	0.1645	0.0046	0.6575	**0.1010**	**0.0037**	**0.6637**	**0.1010**	**0.0037**	**0.6637**
fr1_desk	0.7984	0.0200	0.5985	0.5145	0.0179	0.9084	0.1024	0.0082	0.5961	**0.0792**	**0.0080**	**0.5961**	**0.0792**	**0.0080**	**0.5961**
fr2_desk	1.6240	0.0392	1.2632	1.3240	0.0385	1.0680	0.9729	0.0133	0.4695	**0.9178**	**0.0132**	**0.4713**	**0.9178**	**0.0132**	**0.4713**
fr2_pioneer_360	1.7455	0.2534	2.0123	1.7684	0.0474	1.7020	0.7417	0.0272	0.7302	**0.2534**	**0.0105**	**0.7302**	**0.2534**	**0.0105**	**0.7302**
fr3_walking_static	0.0184	0.0005	0.1917	0.0236	0.0005	0.1817	0.0215	0.0006	0.1982	0.0235	0.7130	0.1566	**0.0226**	**0.3736**	**0.0835**
fr3_walking_xyz	0.2793	0.0049	0.4434	0.2853	0.0047	0.0047	0.2736	0.0045	0.4519	0.2684	1.0564	0.3036	**0.2084**	**0.7294**	**0.2197**
balloon2	0.1711	0.0274	2.5248	0.1665	0.0332	3.3890	0.1355	0.0401	3.6352	0.1448	0.0367	3.5318	**0.0648**	**0.0280**	**2.0559**
balloon_tracking2	0.2683	0.0759	6.6432	0.2885	0.0659	7.7445	0.2793	0.0780	8.5404	0.2877	0.0692	8.6126	**0.2077**	**0.0562**	**7.7626**
crowd3	0.0535	0.0123	2.8557	0.0676	0.0093	2.9696	0.0706	0.0090	3.1230	0.0723	0.0094	3.0891	**0.0323**	**0.0094**	**2.4291**
person_tracking2	0.6632	0.0617	4.7066	0.4305	0.0670	6.0333	0.4907	0.0664	6.4411	0.5492	0.0645	6.3616	**0.4690**	**0.0515**	**5.9916**
placing_nonobs_box	0.1736	0.0236	2.2067	0.1451	0.0167	2.4952	0.1758	0.0169	2.5557	0.1789	0.0177	2.4121	**0.0989**	**0.0161**	**0.6630**
Average	0.5434	0.0480	2.2327	0.4701	0.0283	2.4698	0.3117	0.0244	2.4908	0.2615	0.1820	2.4481	**0.1577**	**0.1806**	**2.7436**

Table 10: Ablation study results.

The Accuracy tends to increase when additional components are integrated with the probabilistic feature-metric DIA.
Every component contributes to enhancing our system.

Discussion and Future Work

Limitation of Proposed Segmentation Approaches (1)

- Geometric segmentation struggles to separate temporally connected objects

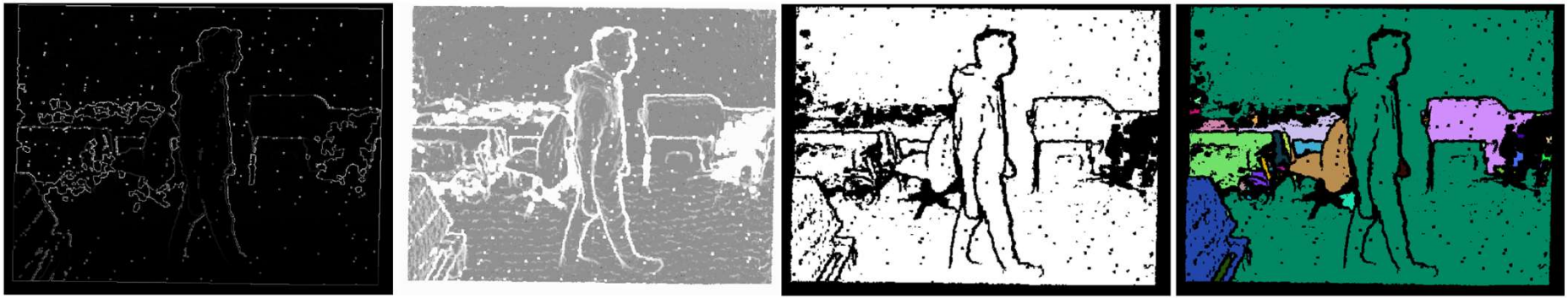


Table 9:Geometric segmentation results on a selected image of the sequence person_tracking from the Bonn RGB-D Dynamic Dataset.

- Experiments:
  - Introduce an additional edge detection term
  - Use deep neural network to generate the edge map

- Segmentation results are improved but introduce new limitations

Discussion and Future Work

Limitation of Proposed Segmentation Approaches (2)

- Generation of prompt points completely relies on the uncertainty map
- Uncertainty maps can be noise, which leads to false positive segmentation

Figure 10. Failure case in uncertainty-based segmentation due to noisy uncertainties

- The long inference time of FastSAM makes it unfeasible for real-time dynamic SLAM
  - ~ 480 ms per frame using TUM RGB-D data

Discussion and Future Work

Other Limitations

- Moving object segmentations are hard-associated with pixels

  → sparse gradient flows

- No back-end optimization

  → cannot overcome accumulated drift over time

Discussion and Future Work

Possible Extension and Future Research

- Improving the tracking accuracy
- Improving the system's efficiency by exploiting CUDA implementation
- More effective segmentation approaches
- Exploiting up-to-date map representation methods, such as NeRF and 3D Gaussian Splatting
- Illustrate the benefits of having a fully differentiable dynamic SLAM pipeline for training downstream robotic tasks

Figure 5. Computational graph of Direct Image Alignment

Summary

- Proposed an automatically differentiable dense visual SLAM pipeline for RGB-D cameras by extending ∇SLAM [1]
- Proposed a novel uncertainty-based moving object segmentation method using FastSAM []
- Our system is capable of estimating camera and object motions and reconstructing the dynamic environment
- The proposed methods achieve promising results, but there are limitations

Thank you!

Questions & Discussion